EXPRESS MAIL LABEL NO.: EV331727012US DATE OF DEPOSIT: September 11, 2003

I HEREBY CERTIFY THAT THIS PAPER IS BEING DEPOSITED WITH THE UNITED STATES POSTAL SERVICE EXPRESS MAIL POST OFFICE TO ADDRESSEE SERVICE UNDER 37 CFR SEC. 1.10 ON THE DATE INDICATED ABOVE AND IS ADDRESSED TO: COMMISSIONER FOR PATENTS, P.O. BOX 1450, ALEXANDRIA, VA 22313-1450; MAIL STOP PATENT APPLICATION.

Karen Orzechowski — Karen Orzechowski
Karen Orzechowski — Signature of Person Mailing Paper

Inventors: Everett A. Corl, Jr. Gordon T. Davis, Clark D. Jeffries, Natarajan Vaidhyanathan, Colin B. Verrilli

APPARATUS AND METHOD FOR CACHING LOOKUPS BASED UPON TCP TRAFFIC FLOW CHARACTERISTICS

BACKGROUND OF THE INVENTION

1. FIELD OF THE INVENTION

The present invention relates to communications networks in general and, in particular, to apparatus and method used to enhance the data management capabilities of devices connected to said network.

2. PRIOR ART

The description which follows presupposes knowledge of network data communications and switches, network processors, adapters, routers, etc. as used in such communications networks. In particular, the description presupposes familiarity with the ISO model of network architecture which divides network operation into layers. A typical architecture based upon the ISO model of network architecture which divides

network operation into layers. A typical architecture based upon the ISO model extends from Layer 1 (also sometimes identified as "L1") being the physical pathway or media through which signals are passed upwards through Layers 2, 3, 4 and so forth to Layer 7, the last mentioned being the layer of applications programming running on a computer system linked to the network. In this document, mention of L1, L2 and so forth is intended to refer to the corresponding layer of a network architecture. The disclosures also presupposes a fundamental understanding of bit strings known as packets and frames in such network communication.

A general model for a communications network may include one or more private networks coupled via a firewall or similar structure to a public network such as the World Wide Web (WWW) better known as the Internet. Communications between devices connected to the network, hereafter called network devices, may occur solely within the private network or through the firewall via the Internet to remote private networks.

In order to exchange information between network devices and to manage the network some type of protocol is required. The protocol could be characterized as a set of rules that govern access to the network and in some cases are used to keep the network in operable condition. Even though there are some standard protocols, such as ethernet, token ring, etc. that can be used on private networks for the most part private network may use any protocol management wishes to use. The only possible restriction is that network devices, on the private network, must be cognizant of the protocol or else the network devices will not be able to operate or communicate satisfactorily.

Because of the lack of uniformity on private protocols further discussion is limited to the public protocol which is used on the internet. The public protocol is referred to as TCP (Transmission Control Protocol)/IP (Internet Protocol). This is a well known protocol which is used to communicate over the internet.

One measure of performance for network devices, such as network processors, running IP routing applications is based upon the number of packets processed or classified within a set time interval such as one second. This in turn can be influenced by the type of classification that is required. For the purpose of routing packets in a network classification may be grouped as Layer 2 (L2), Layer 3(L3), Layer 4(L4) and above. The computational requirements for each of the layers increases from L2 to L4 and above.

For example, L2 classification may be as simple as finding a Media Access Control (MAC) table match. This procedure would require comparing a MAC Source Address (SA) or MAC Destination Address (DA) packet with addresses in a table. A more general approach is to use a Full Match Algorithm, such as the one disclosed in application serial number 09/543531 for L2 classification tasks.

L3 classifications can be used for routing and other L3 functions. L3 classification, if used for routing purposes, requires finding the longest prefix match between information in a packet and information in a database. A Longest Prefix Match Algorithm is used for L3 classification. The Longest Prefix Match Algorithm is more complex than the Full Match Algorithm and, therefore, requires more computational time.

L4 classification includes complex functions such as enforcement of filter rules

with possible complex interceding ranges etc. This type of classification usually requires complex algorithm which use relatively long time interval to process a packet. The time interval to process a packet even increases for lookups or classification above L4. Processing above L4 classification is referred to as deep packet processing.

In view of the above, the throughput (number of packets processed per second) of a network device, such as a network processor, depends on the type of lookups or classifications carried out by the device. As a consequence the computational resources of network processors are stressed when L3 and higher lookups are required. In addition, network processors are often required to carry out lookups with millions of packets per second which further stress the computational resources. With the resources of network processors being stressed, the likelihood of them being able to meet throughput requirements and at the same time process L3 and above lookups appears relatively low. Therefore, a need exists for resources which will enable network devices, such as network processors, to maintain throughput to meet performance requirements and at the same time process L3 and above lookups.

SUMMARY OF THE INVENTION

The present invention provides an accelerator which improves look-up capabilities of network devices and by so doing the network devices are able to classify packets more efficiently than has heretofore been possible.

The accelerator includes a memory called a cache in which characteristics of TCP packets called four-tuple (described herein) are stored. The four-tuple include

Internet Protocol (IP) SA, the IP DA, the Transmission Control Protocol (TCP) source port (SP) and the destination port (DP). Actions associated with each of the four-tuple are also stored. Match logic correlates the four-tuple in a received packet with the four-tuple in the cache. If a match occurs the actions associated with the four-tuple in the catch is applied to the received packet. If a match does not occur the regular process used to classify a packet of that type is followed. Specific methods and apparatus are provided to populate and dynamically age the Flow Cache. The cache and related hardware and/or software are termed "Flow Cache".

By using the Flow Cache a network processor can classify packets at a faster rate than was heretofore possible.

Other object, features and advantages of the present invention will be apparent from the accompanying drawings and from the detailed description which follows below.

BRIEF DESCRIPTION OF THE DRAWINGS

Figure 1 shows a block diagram of a communications network with network devices practicing teachings of the invention.

Figure 2 shows a block diagram of a network processor.

Figure 3 shows a block diagram of a subsystem (Flow Cache and controller) according to the teachings of the present invention.

Figure 4 shows a graphical representation of the format for packets.

Figure 5 shows a format for TCP header.

Figure 6 shows a format for IP header.

Figure 7 shows a flowchart of a program executed on a computer according to teachings of the present invention.

Figure 8 shows a flowchart for purging the Flow Cache according to teachings of the present invention.

DETAILED DESCRIPTION OF THE EMBODIMENT

The present invention uses flow caching to expedite classification of packets in a communications network and hence improve the number of packets the network device can process in a particular unit time. It works well with TCP traffic and as such will be described in that environment. However, this should not be a limitation upon the scope of the invention since it works well with any communication protocol in which traffic occurs in bursts. For example, it is believed that UDP traffic (protocol 17) between DNS servers occur in bursts and could also benefit from flow caching. As a general statement of the invention the general approach is to identify those flows which consistently have the most bundling and only go to the cache for them. Of equal importance is avoiding the cache for those packet types which do not bundle.

Flows or packets with most bundling comes in bursts that occur within relatively short period of time interval and are referred to in this document as "Frequent Flyers".

Figure 1 shows a highly simplified network 100 in which the teachings of the present invention are deployed. The network includes the plurality of edge devices 104 connected to the internet or other network 102. Each of the edge device 104 are connected to a sub network 106 which could be private networks interconnecting users

in a building, campus, etc. Such sub networks are well known in the art and will not be described further. Sub-network 106 is connected by bridge device 110 to another sub network. The edge devices 104 could be a router switch or any other network device used for this type of packet handling. Each of the edge devices is provided with network processor 108. Depending on the type of device one or more network processors could be used. The bridge device 110 could also contain a network processor 108 which is used to perform routing function. The use of network devices such as 104 and 110 are well known in the art, therefore further description of these entities will not be given.

Figure 2 shows a block diagram of the Network Processor shown in Figure 1. The Network Processor executes IP routing applications including those according to teachings of the present invention and classifies frames as they are received from the internet. Different classification tasks including Layer 2 (finding a MAC table match), Layer 3 (finding a longest prefix match for routing), Layer 4 (enforcement of filter rules with possible complicated intersecting ranges of protocol parameters) and higher layer lookup (up to the application layer, e.g. so called deep packet processing) are performed by the Network Processor. Some of the classifications such as Layer 4 classification require relatively long processing times because complex algorithms are involved. The present invention provides an apparatus and technique that significantly reduces the average time necessary to perform Layer 3, Layer 4 and higher layer classification. Even though any type of network processor can be used to perform the classification function in the preferred embodiment of the invention PowerNP developed and marketed by IBM is used. The PowerNP is a single chip device 10 that includes a

plurality of functional units integrated on the substrate. The functional units are arranged into an upside configuration and a downside configuration, with the upside configuration (sometimes also referred to as an "Ingress") referring to those components relating to data inbound to the chip from a data transport network (up to or into the chip) and "downside" (sometimes referred to as an "Egress") referring to those components whose function is to transmit data from the chip toward the data transmission network in an outbound fashion (away from the chip or down and into the network). Data flow follows the respective arrangements of the upside and downside configuration. As a consequence there is an upside data flow and a downside data flow in the system of Figure 2. The upside or Ingress configuration elements include an enqueue dequeue scheduling UP (EDS-UP) logic 16, multiple multiplexed MACs-UP (PMM-UP) 14, Switch data mover up (SDM-DN) 18, system interface SIF(20), data align serial link A (DASL A) 22 and data align serial link B (DASL B) 24. DASL A 22 and DASL B 24 connect the network processor to a switch in systems with many network connections requiring multiple network processors.

Still referring to Figure 2 the components depicted on the down side (or Egress) of the system includes data links (DASL-A) 26 and DASL-B 28, system interface SIF 30, switch data mover (SDM-DM) 32, enqueue-dequeue-scheduler EDS-DN 34 and multiple multiplex MACs for the Egress PMM-DN 36. The network processor chip 10 also includes a plurality of internal static access memory components (S-RAM's), a traffic management scheduler (TRAFFIC MGT SCHEDULER) also known as Egress scheduler 40 and an embedded processor complex 12. The embedded processor complex contains a plurality of processors and coprocessors which are used to perform

the classification function. An interface device 38 is coupled by the respective Data Movement Unit (DMU) buses to PMM 14, and 36. The interface device 38 could be any suitable hardware apparatus for connecting to a network (not shown) such as ethernet physical device or asynchronous transfer mode (ATM) device, etc. A more detailed description of the PowerNP is given in U.S. Patent 6,404,752 which is incorporated in its entirety within this document.

As stated previously the protocol which is used on the internet is the TCP/IP protocol. The present invention is described with respect to data packets using this protocol. The TCP/IP protocol has certain packets referred to in the present invention as "Frequent Flyers" which consistently occur in bursts. Each processor is preprogrammed with rules including actions to be taken relative to packets received, in the Network Processor, and matching a particular rule. According to the teaching of the present invention, Frequent Flyers and their associated actions are placed in the cache and subsequent members of the burst can be processed based upon information stored in the cache. As a consequence the throughput of the system using the present invention is enhanced.

It should be noted that if a match is not found for a packet in the cache the standard classification system and method provided for in the Network Processor is followed. The standard classification system and method requires more time than the cache to process packets. As a consequence the cache reduces the latency associated with standard classification systems.

Figure 4 shows a graphical representation of TCP/IP packet or frame 400 which includes a header portion 402 and a payload 404. The TCP/IP format is well known in

the prior art. Therefore, only the portion of the format which relates to the present invention will be discussed further. The portion of the format which is relevant to the present invention is the header 402. The header section 402 includes both a TCP portion and an IP portion.

Figure 6 shows IP Header format 600 for the IP portion of the header. The IP header format is well known in the prior art. Therefore, only those portions of the header that are of interest to the present invention will be discussed. The portion of the IP header format which is relevant to the present invention is the subfield labeled Protocol, the subfield labeled IP Source Address and the subfield labeled IP Destination Address. As will be discussed subsequently the Source Address (SA) and Destination Address (DA) are two of the four parameters referred to as the four-tuple which will be used as entry into the Flow Cache controller described hereinafter. It should be noted that the Protocol subfield is also used to identify the type of packet that is being transmitted (i.e. TCP).

Figure 5 shows TCP header format 500. The TCP header format is well known in the prior art; therefore, only those fields which are used in the present invention will be discussed. The fields which are used includes the Source Port (SP) and Destination Port (DP). As will be discussed hereinafter the SA, DA from the IP header format and SP, DP from the IP TCP header format are known as the four-tuple which will be used in the Flow Cache. Also of interest are the control keys identified by numeral 502. Each key is a one bit field that identifies the type of packet that is being transmitted. The bit is set by the source device originating the packet. For example if the bit in the key labelled SYN is set to a logical '1' the packet would be a synchronization packet.

The packets identified by 502 are required to establish a session between network devices.

Figure 3 shows a functional block diagram of the Flow Cache system 300 according to the teachings of the present invention. The Flow Cache system includes an associative memory of limited size which stores a mapping between characteristics of flows and action to be taken relative to the flows. In one embodiment of the present invention the characteristics are the four components of the four-tuple, namely, the IP Source Address (SA), the IP Destination Address (DA), the TCP Destination Port (DP) and Source Port (SP). A controller 302 correlates the characteristic (four-tuple) from a received packet with the content of the associated memory. If a match is found, then the actions stored relative to the four-tuple are applied to or imposed on the frame or packet. Extracting information (i.e. action to be taken relative to a packet) from the Flow Cache system 300 is much faster than using the traditional algorithm such as longest prefix match or software managed tree algorithm to determine what action to be taken relative to a packet. It should be noted that even though the description covers Layer 3 and Layer 4 processing, it can also be extended to include Layer 2 or other processing by concatenating N fields from packet headers (one or more layers in the OSI model) to create an "n-tuple". For example, Layer 2 actions could be added to a Layer 2 version of the invention using Layer 2 header fields.

For this invention to be effective the size of the cache has to be controlled. If the size of the cache is too large it could actually degrade performance of the system. The cache should only be large enough to maintain session information for the duration of a burst of the Frequent Flyers. Examination of real Internet traffic shows that this interval

is approximately 1ms. This interval along with the Frequent Flyer packet rate handled by the network processor can be used to compute the required size of the cache.

Ideally, the cache size should be able to be contained within internal fast SRAM and should be small enough to be searched within the time available to process the packet. Note that larger caches may require more time to search.

Figure 7 shows a flowchart of the packet processing algorithm 700 according to the teachings of the present invention. The algorithm is executed in the Network Processor. Preferably, only data frames would be eligible for processing, using the teachings of the present invention, since they typically are transmitted in bursts as a result of fragmenting large messages into multiple smaller packets based on the Maximum-sized Transmission Unit (MTU) supported by the network path. Other packets, such as SYN, FIN or RST, in the TCP/IP protocol suite would not be cached since consecutive packets of these types are typically separated in time by the round trip delay of the required network path. Data packets could be identified as such by examining the flag bits 502 (Figure 5) in the TCP header and possibly the length field in the IP header and the data offset field in the TCP header. The flag bits are also called control bits, both of which are used interchangeably in this document. Those packets which have any of the SYN FIN or RST flags set are not data packets. If none of these bit values are set (i.e. Equal to 1) in the control flag field of the TCP header, then the length field in the IP header must be examined. Next the data offset field in the TCP header (multiplied by 4) must be subtracted from this value. If the result of the subtraction is greater than zero there is data in the packet. Alternatively, some other value greater than 0 could be chosen as a threshold. If the result indicates that there is

data in the TCP packet, then the packet is considered a data packet and is thus a Frequent Flyer. Alternatively, just the length field in the Ethernet header could be compared as a first step to a fixed value such as 1400 Bytes. If greater than the comparison value, then the packet would be considered a data packet.

Referring again to Figure 7 the algorithm begins in start block 702 and descends into 704 where a packet is received. After a packet is received some initial checks are performed in block 706 to see if the packet is a Frequent Flyer and thus cacheable. In block 708 the packet is checked to see if it is a TCP packet by examining the IP and TCP headers. If not, the normal forwarding path for a full packet search is taken (block 710). In the normal forwarding path the packet is forwarded to the full packet search mechanism of the Network Processor. This mechanism could be a full match search algorithm, a longest match prefix algorithm or a software managed tree algorithm. Each of these search algorithms is discussed in Patent Application serial number 09/543,531 (Full Match), Patent Application serial number 09/544,992 (Longest Prefix Match) and Patent Application serial number 09/545,100 (Software Management Tree). The named applications are assigned to the assignee of the present invention and they are incorporated in their entirety herein. With reference to block 708 if the packet is a TCP/IP packet it is checked (block 712) to see if the packet is a data packet (block 714) according to the Frequent Flyer criteria given above. If not, again the normal forwarding path is followed block 716. If it is a data packet (block 714) then a cache lookup is performed using the four-tuple from the packet block 718. If a match is found (block 720) in the cache then the stored actions in the cache are enforced (block 724). The lookups involved in the normal forwarding paths are avoided. If no match was found

block 720 then the normal forwarding path 722 to full packet search is exercised. In addition, a new cache entry (block 726) is added to the cache using the four-tuple of the packet and the action performed on that packet.

It is obvious from the description that the cache works in conjunction with the structure and full packet search algorithms identified above to provide a more efficient packet classification system.

The contents of the cache has to be changed periodically in order for the system to work satisfactorily. This means older entries must be deleted to make space in the cache for newer entries. According to one embodiment, Figure 8 a flowchart 800 of an aging algorithm. The aging algorithm is executed on the Network Processor and periodically delete old entries from the associated memory. The program begins in block 804 and descends into block 805. Block 805 initializes an expiration timer which controls the frequency at which the Flow Cache is aged. This timer would correspond to the expected burst interval of the Frequent Flyers. In block 806, the process waits for a check time interval (smaller than the burst timer) and then decrements the expiration timer value by that check time interval. At this point a check is made for a change in the Layer 3 or Layer 4 database. If a change has been made during the check time interval, then the cache may be invalid and should be purged. In such a case, the cache is purged (block 810). Processing then continues with block 805, starting a new expiration timer.

In the case that the database has not changed (block 808), processing continues with block 812. A check is made for the expiration timer expiring (reaching a value of zero). If the timer has expired, every entry of the cache is checked (block 814). If the

timer has not expired, processing continues with block 806. Each entry which has not been used during the prior expiration interval is removed from the cache (block 816). Processing then continues with block 805, starting a new expiration timer.

In an alternate embodiment, instead of using the aging program 800, algorithm 700 can be modified slightly to perform the aging function. Within block 726, if no space is available for a new cache entry to be added, the least recently used cache entry can be removed and the new entry can be added in its place.

By using the Flow Cache to classify packets in a communications network instead of using traditional algorithm and data structure the time required to classify packets is much shorter and as a result system throughput is enhanced.

The foregoing is illustrative of the present invention and is not to be construed as limiting thereof. Although exemplary embodiments of this invention have been described, those skilled in the art will readily appreciate that many modifications are possible in the exemplary embodiments without materially departing from the novel teaching and advanced use of this invention. Accordingly, all such modifications are intended to be included within the scope of this invention as defined in the claims.